# Consolidating Ranking and Relevance Predictions of Large Language Models through Post-Processing

**Le Yan, Zhen Qin, Honglei Zhuang, Rolf Jagerman,**
**Xuanhui Wang, Michael Bendersky, Harrie Oosterhuis**
Google Research, Mountain View, CA 94043, USA
{lyyanle,zhenqin,hlz,jagerman,xuanhui,bemike,harrie}@google.com

## Abstract

The powerful generative abilities of large language models (LLMs) show potential in generating relevance labels for search applications. Previous work has found that directly asking about relevancy, such as "*How relevant is document A to query Q?*", results in sub-optimal ranking. Instead, the pairwise-ranking prompting (PRP) approach produces promising ranking performance through asking about pairwise comparisons, e.g., "*Is document A more relevant than document B to query Q?*". Thus, while LLMs are effective at their ranking ability, this is not reflected in their relevance label generation.

In this work, we propose a post-processing method to consolidate the relevance labels generated by an LLM with its powerful ranking abilities. Our method takes both LLM generated relevance labels and pairwise preferences. The labels are then altered to satisfy the pairwise preferences of the LLM, while staying as close to the original values as possible. Our experimental results indicate that our approach effectively balances label accuracy and ranking performance. Thereby, our work shows it is possible to combine both the ranking and labeling abilities of LLMs through post-processing.

## 1 Introduction

Generative large language models (LLMs) have shown significant potential on question answering and other conversation-based tasks (OpenAI, 2023; Google et al., 2023) owing to their extraordinary generative abilities and natural language understanding capabilities. Naturally, previous work has further investigated the application of LLMs to other areas, including search and recommendation tasks (Zhu et al., 2023; Wu et al., 2023). The goal here is to rank items according to their relevance to a certain query. Generally, existing approaches have applied LLMs to this task in two different ways: First, as pseudo-raters, LLMs are asked to simulate human raters by generating a relevance label for each query-document pair (Liang et al., 2022), for example, through prompts such as "*How relevant is document A to query Q?*" Secondly, an LLM can also be asked directly about the ordering of documents for a query. For example, the pairwise-ranking-prompting (PRP) method (Qin et al., 2023) uses a prompt like "*Is document A more relevant than document B to query Q?*" Alternatively, LLMs can be asked to generate the entire ranking through a prompt like "*Rank the following documents by their relevance to query Q: document A, document B, document C, etc.*" (Sun et al., 2023a) Thus, there are several distinct modes by which LLMs can be used for ranking purposes, which provide different kinds of output.

Each mode of applying LLMs to ranking tasks offers distinct advantages in terms of performance and efficiency. The pseudo-rater mode is currently favored in LLM applications within ranking systems due to its simplicity and high efficiency (Liang et al., 2022; Sachan et al., 2022; Thomas et al., 2023; Oosterhuis et al., 2024). Given the high costs associated with deploying or training LLMs for high-throughput applications like search and recommendations, it is, so far, only efficiently feasible to use LLMs as pseudo-raters to label a fraction of raw data in zero-shot or few-shot fashion as a replacement of more expensive human raters. However, the general LLMs are not tuned to generate meaningful ranking scores, as a result, there is still an apparent gap between state of the art (SOTA) ranking performance and the performance reached when leveraging LLM pseudo-labels for model training (Thomas et al., 2023).

In parallel to exploring the costly fine-tuning of LLMs as ranking specialists (Nogueira et al., 2020; Zhuang et al., 2023b), previous work has also investigated the direct ranking modes of LLMs, where no finetuning is involved. Some of these direct



*Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 410–423

ranking modes, such as PRP (Qin et al., 2023), can reach SOTA ranking performance that is on-par with LLMs finetuned for ranking. Moreover, PRP enables open-source (OSS) LLMs to outperform the largest commercial models like GPT-4 (OpenAI, 2023). However, document scoring by PRP solely considers the resulting order of the candidate list, and thus, the absolute values of scores are meaningless. This makes PRP results unsuitable to be directly used as pseudo-labels. For example, the PRP ranking score of a fair candidate in the list of only poor candidates would be comparable to that of a good candidate in the list of strong competing candidates, (see example in Figure 1). How to effectively combine these direct ranking modes with the pseudo-rater mode to consolidate ranking and relevance predictions of LLMs remains an essential challenge in applying LLMs to real world main stream applications.

In this work, we study post-processing methods to do the consolidation, especially for the case when we have no human labelled data. We first define the problem in LLM ranking in Section 3, and propose our post-processing methods to consolidate LLM predictions for unlabelled data in Section 4. We discuss our experiments on public ranking datasets in Section 5 and show our methods could approach the state of the art ranking performance with minimal tradeoff in relevance prediction performance in Section 6. Our contributions include:

- The first systematic study on the tradeoff between ranking and relevance predictions of LLMs.
- A ranking-aware pseudo-rater pipeline with a novel post-processing method using constrained regression to combine both PRP ranking and LLM relevance generation.
- Extensive experimental study on public ranking datasets that demonstrates the effectiveness of our proposed methods.

## 2 Related Work

The strong capability of LLMs in textual understanding has motivated numerous studies leveraging LLM-based approaches for textual information retrieval (Bonifacio et al., 2022; Tay et al., 2022b; Jagerman et al., 2023). Before the generative LLM era, the focus was more on finetuning pre-trained language models (PLMs) such as T5 (Nogueira et al., 2020; Zhuang et al., 2023b) or BERT (Nogueira and Cho, 2019) for the supervised learning to rank problem (Liu, 2009; Qin et al., 2021), which becomes less feasible with larger generative LLMs. Two popular methods—-relevance generation (Liang et al., 2022; Zhuang et al., 2023a) and query generation (Sachan et al., 2022)-—aim to generate per-document relevance scores or retrieval queries using generative LLMs. These methods are also termed pointwise approaches for ranking. More recent works (Sun et al., 2023a; Ma et al., 2023; Pradeep et al., 2023; Tang et al., 2023) explore listwise ranking generation approaches by directly inserting the query and a list of documents into a prompt. Pairwise order generation through pairwise prompts (Qin et al., 2023) turns out to be very effective for ranking purposes, especially for moderated-sized LLMs. However, none of these ranking approaches using generative LLMs attempt to consolidate the results with relevance generation.

Previous works on non-LLM neural rankers (Yan et al., 2022; Bai et al., 2023) focus on balancing or aligning regression with ranking objectives during the model training, which is unfortunately not feasible for LLMs using zero-shot or few-shot prompting. Post-processing methods that calibrate model predictions using some validation data could be potentially applicable. Originally developed for classification model calibration (Menon et al., 2012), these methods include parametric approaches like Platt scaling (Platt, 2000) for binary classification; piecewise linear transformation (Ravina et al., 2021) for regression; and non-parametric approaches like isotonic regression (Menon et al., 2012; Zadrozny and Elkan, 2002), histogram binning, and Bayesian binning (Zadrozny and Elkan, 2001; Naeini et al., 2015). But how effectively these post-processing approaches could be extended to LLM-based ranking and relevance predictions has not been well studied in existing literature.

## 3 Problem Formulation

We formulate the problem of consolidating ranking and relevance predictions within this framework. Given a set of queries, for each query $q$, we have a set of corresponding candidate documents $\{d\}_q$, and their ground truth labels, $\{y\}_q$, as their relevance evaluations, such as graded relevance. Our first goal is to predict the relevance labels based on the content of each corresponding candidate.

Our second goal is to predict a ranked list of candidates, and we use $\{r\}_q$ to denote the rank of each candidate in this predicted ranking. The predicted ranking is optimal when the ranks align with the order of the relevance labels: $r_i \leq r_j$ if $y_i \geq y_j$ for any pair of candidates $(d_i, d_j)$ belonging to the same query $q$. Taken together, our overall task is to optimize LLM predictions for both relevance estimation and ranking performance.

### 3.1 Relevance Prediction

For this purpose, in this work, we consider real-number predictions, i.e., $\hat{y}_i \in \mathbb{R}$, as the relevance pseudo-labels for query-document pairs. Such pointwise real-number ratings can be averages over the annotations of multiple human raters. For LLM-based raters, pseudo-labels can be obtained from the average rating of raters with discrete output space (Thomas et al., 2023) or from finer-grained rating generation (Zhuang et al., 2023a), or directly leveraging the token probabilities to formulate the relevance predictions if available in the generative LLMs (Liang et al., 2022).

In specific, we use LLM as a rater to generate "Yes" or "No" to answer the question "does the passage answer the query?" for each query-document pair. See Appendix A.1 for the prompt. We obtain the generation probabilities $P_i(\text{Yes})$, $P_i(\text{No})$ and take

$$\hat{y}_i = \frac{P_i(\text{Yes})}{P_i(\text{Yes}) + P_i(\text{No})} \tag{1}$$

as the normalized relevance prediction: $\hat{y}_i = 1$ for the most relevant document and $\hat{y}_i = 0$ for the least.

To evaluate the relevance prediction performance of $\{\hat{y}\}_q$, we consider the mean squared error (MSE):

$$\text{MSE}(\{y\}_q, \{\hat{y}\}_q) = \frac{1}{|\{d\}_q|} \sum_{i \in \{d\}_q} (\hat{y}_i - y_i)^2, \tag{2}$$

as well as the empirical calibration error (ECE) (Naeini et al., 2015; Guo et al., 2017):

$$\text{ECE}_q = \frac{1}{|\{d\}_q|} \sum_{m=1}^{M} \left| \sum_{i \in B_m} y_i - \sum_{i \in B_m} \hat{y}_i \right|, \tag{3}$$

where we group candidates of each query into $M$ successive bins of model score-sorted results $B_m$, and $|\{d\}_q|$ gives the size of candidate documents to query $q$. Compared to MSE, ECE is more sensitive to the distribution divergence between predictions and ground truth labels due to binning.

### 3.2 Ranking Prediction

In the pairwise ranking prompting (PRP) mode, LLMs generate pairwise preferences: for any two documents $d_1$ and $d_2$, LLMs are prompted to generate "$d_1$" or "$d_2$" to answer the question on "which of the passages is more relevant to the query?" See Appendix A.2 for the prompt. Based on the results and the consistency of results when switching the order of $d_1$ and $d_2$ in the prompt, we could have $d_1$ consistently better ($d_1 > d_2$), $d_2$ consistently better ($d_1 < d_2$), and inconsistent judgement ($d_1 = d_2$), as the LLM generated preferences.

To get a consistent ranking from these pairwise preferences, we follow Qin et al. (2023) to compute a ranking score $s_i$ for each document $d_i$ by performing a global aggregation on all other candidates of the same query,

$$\hat{s}_i = 1 \times \sum_{j \neq i} \mathbb{I}_{d_i > d_j} + 0.5 \times \sum_{j \neq i} \mathbb{I}_{d_i = d_j}, \tag{4}$$

where $\mathbb{I}_{cond}$ is an indicator function of the condition $cond$: 1 when cond is true and 0 otherwise. $\hat{s}_i$ essentially counts number of wins for each document. We then sort the candidates by their ranking scores $\{\hat{s}\}_q$ to get predicted ranking $\{r\}_q$.

The ranking performance is evaluated by the normalized discounted cumulative gain (NDCG) metric:

$$\text{DCG}_q = \sum_{i \in \{d\}_q} \frac{2^{y_i} - 1}{\log_2(1 + r_i)}, \tag{5}$$

$$\text{NDCG}_q = \frac{1}{\text{DCG}_q^{\text{ideal}}} \text{DCG}_q, \tag{6}$$

where $\text{DCG}_q^{\text{ideal}} = \max_{\{r\}_q} \text{DCG}_q$ is the optimal DCG obtained by sorting documents by their labels (Järvelin and Kekäläinen, 2002). In practice, the NDCG@$k$ metric that cuts off at the top $k$ results is used.

### 3.3 The Consolidation Problem

Although the two formulations, relevance and ranking predictions, are conceptually aligned to the same ground-truth labels, different modes above are leveraged in practice for different purposes: the pseudo-rater mode of LLMs, directly predicting the candidate relevance to a query, gives relatively good relevance estimation $\hat{y}$ (Liang et al., 2022), while the ranker mode of LLMs, using pairwise prompting, achieves significantly better NDCG but with totally uncalibrated ranking scores $\hat{s}$ that have

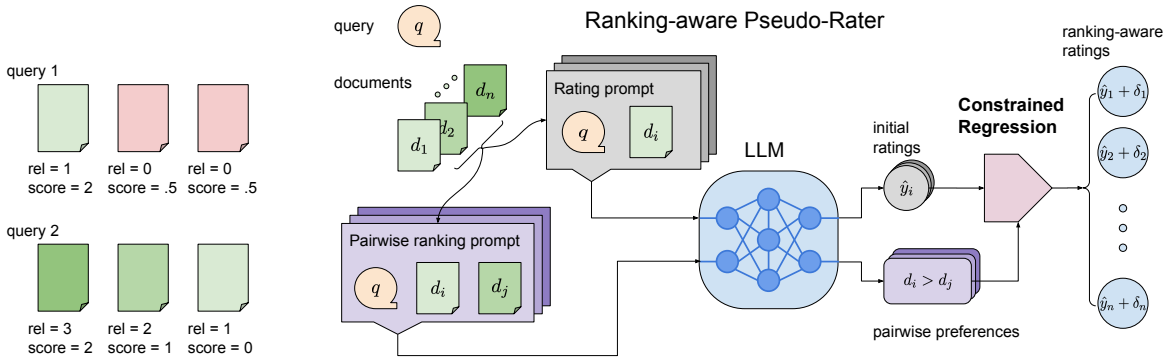


Figure 1: Left: Example of PRP scores not calibrated over different queries. Right: Illustration of the ranking-aware pseudo-rater pipeline that generates ranking-aware ratings with LLMs from the input query and list of candidate documents.

poor relevance prediction performance (Qin et al., 2023), or see Figure 1 for an example. How to address this dichotomy then is the problem that we study in this paper.

In the optimization problem with multiple objectives like this, optimizing for both relevance prediction and ranking performance, the success is difficult to be measured with a single metric. Additionally, a tradeoff typically exists between these metrics (ECE and NDCG in our case) – improving one leading to demoting the other, represented by a Pareto front in the figure of both metrics. Please see examples in Figure 3. An improvement against the baselines is qualified by whether the new method could push the Pareto front by positing metrics on the better side of the current Pareto front.

## 4 The Methods

This section presents our post-processing methods to consolidate the ranking scores $\hat{s}$ as well as the pairwise preferences from the LLM ranker mode and the relevance estimation $\hat{y}$ from the pseudo-rater mode, aiming to optimally balance ranking and relevance prediction performance. To make a fair comparison with previous LLM rankers, we stick to zero-shot prompting results with no training or finetuning.

Specifically, we introduce a constrained regression method to find minimal perturbations of the relevance predictions $\hat{y}$ such that the resulting ranking matches the the pairwise preference predictions of PRP. Additionally, we also introduce an efficient version of our constrained regression method that avoids querying an LLM to construct the complete quadratic number of pairwise constraints by selecting a linear-complexity subset of pairwise comparisons. Finally, with the constrained regression to consolidate, we propose a ranking-aware pseudo-rater pipeline that leverages both rating and ranking capabilities of LLMs to make high-quality ratings for search.

### 4.1 Constrained Regression

The goal of the constrained regression methods is to adjust the LLM relevance predictions $\hat{y}$ so that their order aligns with the ranking order of the PRP results $\hat{s}$. By minimizing the perturbations to adjust the predictions, the resulting scores should closely match the original relevance predictions while adhering to the PRP's ranking performance.

Formally, given a query $q$, we aim to find a set of minimal linear modifications $\{\delta\}_q$ of the LLM relevance predictions, so that for a PRP pairwise preference $d_i > d_j$ or $\hat{s}_i > \hat{s}_j$, the modified predictions match that order: $\hat{y}_i + \delta_i > \hat{y}_j + \delta_j$. In general terms:

$$\{\delta^*\}_q = \text{argmin}_{\{\delta\}_q} \sum_{i \in \{d\}_q} \delta_i^2 \quad (7)$$
$$s.t.\ \Delta_{ij}[(\hat{y}_i + \delta_i) - (\hat{y}_j + \delta_j)] \geq 0$$
$$\text{for} \quad \forall i, j \in \{d\}_q,$$

where $\Delta_{ij} = \hat{s}_i - \hat{s}_j$, if preference is constructed from ranking scores, or $\Delta_{ij} = \mathbb{I}_{d_i > d_j} - \mathbb{I}_{d_i < d_j}$ if direct preference is considered. Thus, the sign of $\Delta_{ij}$ indicates the pairwise order between $i$ and $j$, and a lack of preference in ordering results in $\Delta_{ij} = 0$. We use $\{\hat{y} + \delta^*\}$ as our final predictions for both ranking and relevance.

The mathematical problem posed in Eq. 7 is a well-known constrained regression problem that can easily be solved with publicly available existing math libraries (Virtanen et al., 2020).

### 4.2 Efficiency Improvements

Constrained regression is a traditional, fast, and cost-efficient algorithm compared to LLM opera-

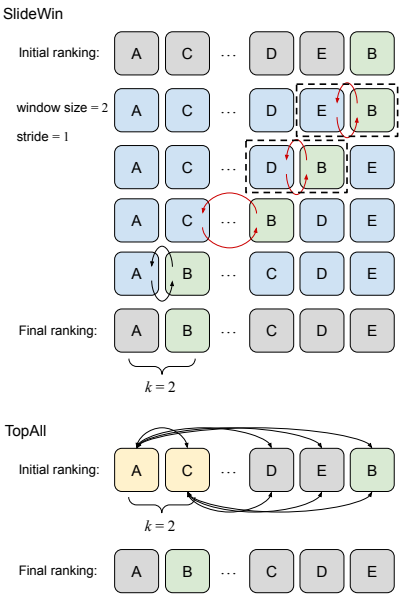


Figure 2: Illustration of how to select LLM pairwise constraints in SlideWin and TopAll methods. Top: SlideWin method with window size 2 and stride 1 takes $o(kn)$ successive pair comparisons, illustrated by paired arrows, to sort for top $k$ results from some initial ranking. Bottom: TopAll method considers top-$k$ results from an initial ranking and their pairwise constraints with all other results, shown by $o(kn)$ double-headed arrows.

tions, as detailed in Section B. A limitation of the above method is the need to identify all $o(n^2)$ pairwise constraints through pairwise ranking prompting to calculate ranking scores $\hat{s}$ in Eq. 4 for a list of size $n$. As the method only depends on pairwise constraints given by $\Delta_{ij}$, a simple way to improve efficiency is to reduce the number of pair constraints to be processed by LLM.

Here we introduce two efficient constraint choices: SlideWin and TopAll, as illustrated in Figure 2. (1) As the ranking performance focuses mostly on the top results (top 10 or top 20), PRP work (Qin et al., 2023) proposes to just run a sliding window sorting from some initial ranking to find the top-$k$ results with $o(kn)$ pair comparisons. We just reuse these $o(kn)$ pair comparisons as constraints $\Delta_{ij}$ in Eq. 7. We call this variant SlideWin. (2) As our final predictions rely upon the relevance scores $\hat{y}$, we don't need to sort from random. Assuming the initial ranking from initial relevance scores $\hat{y}$ is close to the final PRP ranking, we can just consider pairwise constraints between the candidates of top relevance predictions and the rest. In specific, we consider top-$k$ in the relevance scores $\hat{y}$ and all other results in the candidate list, or top-$k$ vs. all, where $o(kn)$ pair constraints to be enforced. We call this variant TopAll.

Table 1: Summary of constrained regression methods vs Pseudo-Rater and PRP baselines.

| Methods | Use $\hat{y}$ | Use $\{d_i > d_j\}$ | Complexity of LLM calls |
|---|---|---|---|
| PRater | Yes | No | $o(n)$ |
| PRP | No | Yes, all | $o(n^2)$ |
| Allpair | Yes | Yes, all | $o(n^2)$ |
| SlideWin | Yes | Yes, partial | $o(n)$ |
| TopAll | Yes | Yes, partial | $o(n)$ |

In Table 1, we summarize the use of LLM-generated relevance predictions $\hat{y}$ and pairwise preferences $\{d_i > d_j\}$ and the method complexities in terms of LLM calls of all proposed methods together with the Pseudo-rater and PRP baselines. More efficiency analysis can be found in Appendix B.

### 4.3 Ranking-Aware Pseudo-Rater

To conclude, we propose a ranking-aware pseudo-rater pipeline that leverages both the rating and ranking capabilities of LLMs, as illustrated in Figure 1. For a given query $q$ and a list of candidate documents $\{d\}_q$, we formulate pointwise rating and pairwise ranking prompts, then feed these prompts to the central LLM to obtain initial ratings and pairwise preferences, respectively. We then combine the initial ratings and pairwise preferences using our constrained regression methods for consolidation. The output of this pipeline is the ranking-aware pseudo labels.

## 5 Experiment Setup

We conduct experiments using several public ranking datasets to answer the following research questions:

- **RQ1**: Can our proposed constrained regression methods effectively consolidate the ranking performance of PRP and the relevance performance of LLMs as psuedo-raters?
- **RQ2**: What is the tradeoff between ranking and relevance prediction performance for different methods?

### 5.1 Datasets

We consider the public datasets with multi-level labels to study the above research questions. Specifically, we utilize the test sets of TREC-DL2019

Table 2: Statistics of experimental datasets.

| Dataset | # of queries | labels | normalized labels |
|---|---|---|---|
| TREC-DL2019 | 43 | {0, 1, 2, 3} | {0, 1/3, 2/3, 1} |
| TREC-DL2020 | 54 | {0, 1, 2, 3} | {0, 1/3, 2/3, 1} |
| TREC-Covid | 50 | {0, 1, 2} | {0, 1/2, 1} |
| DBPedia | 400 | {0, 1, 2} | {0, 1/2, 1} |
| Robust04 | 249 | {0, 1, 2} | {0, 1/2, 1} |

and TREC-DL2020 competitions, as well as those from TREC-Covid, DBPedia, and Robust04 in the BEIR dataset (Thakur et al., 2021). Table 2 summarizes the statistics of queries and the range of labels. The candidate documents are selected from the MS MARCO v1 passage corpus, which contains 8.8 million passages. LLM rankers are applied on the top 100 passages retrieved by BM25 (Lin et al., 2021) for each query, same setting as existing LLM ranking works (Sun et al., 2023a; Ma et al., 2023; Qin et al., 2023).

### 5.2 Evaluation Metrics

For ranking performance, we adopt NDCG (as defined in Eq. 5) as the evaluation metric, with higher values indicating better performance. We primarily focus on NDCG@10, but also present NDCG with other cutoff points in certain ablation studies. For the relevance prediction performance, we use the *mean squared error* (MSE) in Eq. 2 and the *empirical calibration error* (ECE) in Eq. 3 as the evaluation metrics. The lower ECE values indicate better relevance predictions. In this work, we choose $M = 10$ bins (Naeini et al., 2015) with each bin containing approximately the same number of documents ($\sim 10$ documents per bin).

### 5.3 Comparison Methods

We investigate the performance of the following methods in ranking and relevance prediction:

- **BM25** (Lin et al., 2021): The sole non-LLM ranker baseline.
- **PRater** (Sun et al., 2023a): The pointwise LLM relevance pseudo-rater approach.
- **PRP** (Qin et al., 2023): The LLM ranker using pairwise ranking prompting (PRP). All pair comparisons are used to compute the ranking scores (as in Eq. 4).
- **Allpair** (Ours): The naive constrained regression method in Eq. 7 with all pairwise preferences based on the PRP scores, $\Delta_{ij} = \hat{s}_i - \hat{s}_j$.
- **SlideWin** (Ours): The constrained regression method in Eq. 7 with pairwise LLM constraints collected with the sliding window ordering approach, proposed by Qin et al. (2023): pair comparisons are selected from sliding bottom up on the initial order by BM25 scores with sliding window size $k = 10$.
- **TopAll** (Ours): The constrained regression method with pairwise LLM constraints on the pairs between top $k = 10$ results by sorting on pseudo-rater predictions $\hat{y}$ versus all candidates in the list.

Unless specified, all LLM results in above methods are based on the FLAN-UL2 model (Tay et al., 2022a), an OSS LLM [1].

In addition, motivated by the multi-objective approach to consolidate ranking and relevance predictions in non-LLM rankers (Yan et al., 2022), we also consider a simple weighted ensemble of PRater predictions $\hat{y}$ and PRP scores $\hat{s}$:

$$\hat{y} + w\hat{s}, \tag{8}$$

where $w$ is the relative weight, and we use **Ensemble** to refer the method. Note that in practice some labeled data is needed to decide $w$, while the other methods discussed above are fully unsupervised.

### 5.4 Prediction Normalization

It should be noted that none of the methods are optimized for ground truth label values, hence, the ECE and MSE metrics from the raw results are not directly comparable. Thus, we scale their predictions to match the range of the ground truth labels:

$$\tilde{y} = y_{\min} + (y_{\max} - y_{\min}) \frac{\hat{y} - \min(\hat{y})}{\max(\hat{y}) - \min(\hat{y})}, \tag{9}$$

where $\max$ and $\min$ are global max and global min on the full test set. Subsequently, we compute ECE based on the scaled predicted scores $\tilde{y}$. For normalized relevance labels, we insert $y_{\min} = 0$ and $y_{\max} = 1$.

### 5.5 Supervised PWL Transformation

We also compare a post-processing method *requiring labelled data*, specifically the *piecewise linear transformation* (PWL) introduced in Ravina et al.

[1] https://huggingface.co/google/flan-ul2

Table 3: Evaluation of LLM-based ranking methods on both ranking (NDCG@10) and relevance prediction (ECE and MSE) metrics on TREC-DL 2019 and 2020, TREC-Covid, DBPedia, and Robust04. Bold numbers are the best of all and numbers underlined are the best among proposed methods in each row. Upscript "†" indicate statistical significance with p-value=0.01 of better performance than the baselines, PRater for NDCG@10 and PRP for ECE and MSE.

| | | Baselines | | | | | Our Consolidation Methods | | |
|---|---|---|---|---|---|---|---|---|---|
| | Method | BM25 | PRater | PRP | PRater+PWL | PRP+PWL | Allpair | SlideWin | TopAll |
| TREC-DL2019 | NDCG@10 | 0.5058 | 0.6461 | 0.7242 | 0.6461 | 0.7242 | 0.7236† | **0.7265**† | 0.7189† |
| | ECE | 0.2088 | 0.1167 | 0.3448 | 0.1199 | 0.1588 | **0.1084**† | 0.1090† | 0.1199† |
| | MSE | 0.1096 | 0.0688 | 0.1787 | 0.0652 | 0.0836 | **0.0592**† | 0.0601† | 0.0692† |
| TREC-DL2020 | NDCG@10 | 0.4796 | 0.6539 | **0.7069** | 0.6539 | **0.7069** | 0.7054† | 0.7046† | 0.7025 |
| | ECE | 0.2219 | 0.0991 | 0.3690 | **0.0793** | 0.0954 | 0.0865† | 0.0911† | 0.0966† |
| | MSE | 0.1122 | 0.0632 | 0.1978 | **0.0444** | 0.0488 | 0.0519† | 0.0560† | 0.0600† |
| TREC-Covid | NDCG@10 | 0.5947 | 0.7029 | **0.8231** | 0.7029 | **0.8231** | 0.8220† | 0.7943† | 0.7962† |
| | ECE | 0.2460 | 0.2047 | 0.2340 | **0.1590** | 0.2192 | 0.1990† | 0.1984† | 0.2216 |
| | MSE | 0.2268 | 0.1756 | 0.1621 | **0.1419** | 0.1557 | 0.1575† | 0.1644 | 0.1870 |
| DBPedia | NDCG@10 | 0.3180 | 0.3057 | 0.4613 | 0.3057 | 0.4613 | 0.4598† | **0.4651**† | 0.4029† |
| | ECE | 0.2183 | 0.1360 | 0.4364 | **0.0554** | 0.0629 | 0.1302† | 0.1308† | 0.1329† |
| | MSE | 0.0864 | 0.0967 | 0.2571 | 0.0387 | **0.0350** | 0.0846† | 0.0863† | 0.0901† |
| Robust04 | NDCG@10 | 0.4070 | 0.5296 | **0.5551** | 0.5296 | **0.5551** | 0.5532† | 0.5364 | 0.5347 |
| | ECE | 0.1291 | **0.0650** | 0.4154 | 0.0689 | 0.0658 | 0.0654† | 0.0669† | 0.0804† |
| | MSE | 0.0594 | 0.0386 | 0.2285 | 0.0368 | **0.0361** | 0.0379† | 0.0390† | 0.0509† |

(2021), defined as follows,

$$f\big(s \mid \{\tilde{s}_m, \tilde{y}_m\}_{m=1}^{M}\big) = \tag{10}$$
$$\begin{cases} \tilde{y}_1 & s \le \tilde{s}_1, \\ \tilde{y}_m + \frac{\tilde{y}_{m+1}-\tilde{y}_m}{\tilde{s}_{m+1}-\tilde{s}_m}(s-\tilde{s}_m) & \tilde{s}_m < s \le \tilde{s}_{m+1}, \\ \tilde{y}_M & s > \tilde{s}_M, \end{cases}$$

where $\{\tilde{s}_m, \tilde{y}_m\}_{m=1}^{M}$ are $2M$ fitting parameters. $\tilde{y}_{m+1} > \tilde{y}_m$ and $\tilde{s}_{m+1} > \tilde{s}_m$ are enforced for any $m$ to reinforce the monotonicity of the transformation to effectively scale predictions without affecting the ranking order.

We apply PWL to baseline methods PRater and PRP as a special set of baselines with labelled data available, named as **PRater+PWL** and **PRP+PWL** in the results. Comparing these with supervised methods allow for a better understanding of our proposed unsupervised approaches. To compute the post-fitting in PWL, we apply four-fold cross-validation to the test set data: we randomly divide the test set into four folds by queries, and then fit the PWL transformation function on one set and predict on one of the others, repeatedly, to get PWL transformation results for the whole test set.

## 6 Experimental Results

### 6.1 Main Results

The main results, summarized in Table 3 and Figure 3, include the following observations:

- MSE and ECE metrics are consistent in Table 3. Therefore, we will focus on ECE for the remainder of the discussion.
- Without PWL transformations, the pointwise relevance LLM rater (PRater) performs better in labelling than both the naive BM25 and PRP rankers, as evidenced by a consistenly lower ECE in Table 3.
- Despite its poor ECE, PRP has the best or nearly best ranking performance in terms of NDCG.
- The constrained regression approach can best leverage the relevance estimations of PRater and the ranking capability of PRP and reaches comparable ranking performance in terms of NDCG to PRP, and on par or even better relevance prediction in terms of ECE to PRater.
- Our methods consolidate the ranking from PRP and relevance predictions from PRater effectively, evident by that the combined performance on NDCG and ECE sits well beyond the Pareto fronts of simple weighted Ensemble of the two.
- Our consolidation methods even outperform PRP+PWL, the one with extra data, in ECE on 4 out of 5 datasets and while keeping ranking performance in NDCG@10 as good on all datasets. This is because supervised methods may not learn effectively with limited annotations, which

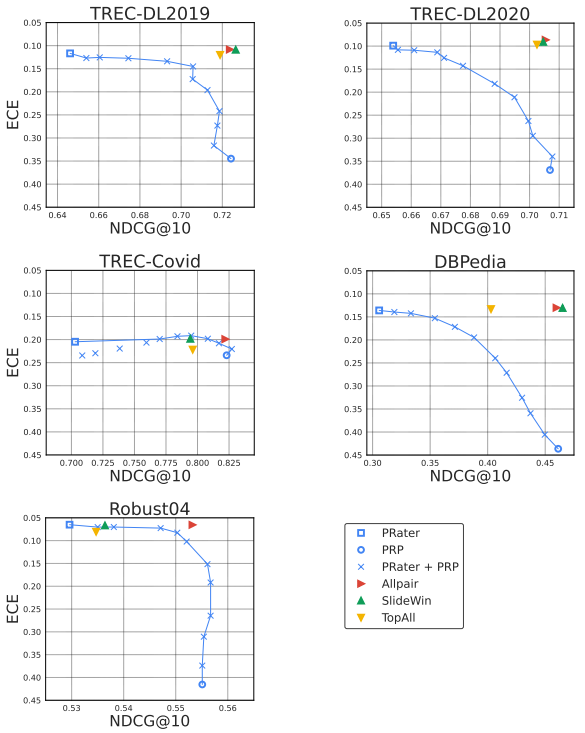


Figure 3: Tradeoff plots on ECE versus NDCG@10 on five ranking datasets. NDCG@10 is higher the better and ECE is lower the better. Overall better methods are on the top right corner of the plots. Lines correspond to the Pareto fronts of Ensemble of PRater and PRP by tuning the weight $w$ in Eq. 8. Our consolidation methods in Table 3 are scattered in the Figure.

is the case for public search datasets given the high cost of collecting human annotations.

- Finally, efficient constrained regression methods may trade off some performance in ranking and regression for the efficiency, but they can still outperform the baselines of PRater and PRP and weighted ensemble of the two in most of the datasets.

With these main results, we can answer the main research questions. **RQ1.** Using the constrained regression methods, we can boost the LLM raters with the superior ranking capability of PRP rankers while keep their relevance predictions nearly untouched. **RQ2.** Naive ensemble of LLM pseudo-rater predictions and PRP scores may lead to a tradeoff between ranking and relevance prediction performance. However, we can get over this tradeoff with the constrained regression methods.

### 6.2 Model Size Effect

As with other tasks involving pretrained LLMs, larger models generally perform better in both ranking and regression metrics. We studied the size effect by comparing results of the FLAN-UL2 model (20B parameters) with those of the FLAN-T5-XXL [2] model (11B parameters). Table 4 shows that our constrained regression methods achieve significantly better NDCG, and comparable or better ECE with the FLAN-UL2 model compared to the FLAN-T5-XXL model. The same size effect is observed in PRater and PRP as well. This shows our consolidation method scales together with the underlying LLM's performance.

Table 4: Model size effect of constrained regression methods and LLM baselines on TREC-DL 2020. The better metrics of the two sizes are bolded per method.

| | NDCG@10 | | ECE | |
|---|---|---|---|---|
| Method | T5-XXL | UL2 | T5-XXL | UL2 |
| PRater | 0.6258 | **0.6539** | **0.0949** | 0.0991 |
| PRP | 0.6985 | **0.7069** | 0.3698 | **0.3690** |
| Allpair | 0.6960 | **0.7054** | 0.0871 | **0.0865** |
| SlideWin | 0.6735 | **0.7046** | **0.0900** | 0.0911 |
| TopAll | 0.6794 | **0.7025** | 0.1038 | **0.0966** |

We have also run experiments on the choices of initial ranking models and choices of parameter $k$ for efficient constrained regression methods (SlideWin and TopAll). The results are included in Appendix C.

## 7 Conclusion

In this work, we have studied the problem of consolidating ranking and relevance predictions of LLMs. We have found that the direct scores from the zero-shot pairwise ranking prompting (PRP) poorly correlate with ground truth labels. To leverage the superior ranking ability of PRP while aligning closely with the ground truth labels, we have investigated post-processing methods and proposed a class of constrained regression methods that combine pointwise ratings from the LLM raters and pairwise constraints from the PRP rankers to take advantage of the two. We have demonstrated with experiments on public ranking datasets that our methods are efficient and effective, offering competitive or superior ranking performance compared to the PRP baseline and relevance prediction performance akin to the pointwise LLM rater. Last but not least, we have proposed a novel framework on how to effectively use generative LLMs to generate ranking-aware ratings, foundation for LLM-powered search ranking.

[2] https://huggingface.co/google/flan-t5-xxl

## Limitations

First, our work mainly focused on consolidating relevance raters with pairwise LLM rankers due to their effectiveness, particularly with moderate-sized open-sourced LLMs. Our methods can be applied to listwise ranking results from listwise LLM rankers (Sun et al., 2023b) by decomposing their ranking results into pairwise comparisons. Our results can be found in Appendix D. However, more effective methods to consolidate listwise rankers, may exist, which we consider for future work. Second, our framework assumes reasonable rating and ranking performance by LLMs. Although generally supported by advances in LLM research and validated across diverse datasets, more advanced adjustments may be required for scenarios where LLMs perform suboptimally, such as in domains opaque to the underlying LLMs.

## References


Aijun Bai, Rolf Jagerman, Zhen Qin, Le Yan, Pratyush Kar, Bing-Rong Lin, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2023. Regression compatible listwise objectives for calibrated ranking with binary relevance. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4502–4508.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. InPars: Unsupervised dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2387–2392.

Google, Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. 2023. PaLM 2 technical report.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query expansion by prompting large language models. *arXiv preprint arXiv:2305.03653*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundation and Trends® in Information Retrieval*, 3(3):225–331.

Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-shot listwise document reranking with a large language model. *arXiv preprint arXiv:2305.02156*.

Aditya Krishna Menon, Xiaoqian Jiang, Shankar Vembu, Charles Elkan, and Lucila Ohno-Machado. 2012. Predicting accurate probabilities with a ranking loss. In *Proceedings of the 29th International Conference on Machine Learning*, pages 703–710.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pre-trained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718.

Harrie Oosterhuis, Rolf Jagerman, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2024. Reliable confidence intervals for information retrieval evaluation using generative ai. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2307–2317.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

John Platt. 2000. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In Alexander J. Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans, editors, *Advances in Large Margin Classifiers*, page 61–74. MIT Press.

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. Rankzephyr: Effective and robust zero-shot listwise reranking is a breeze! *arXiv preprint arXiv:2312.02724*.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, et al. 2023. Large language models are effective text rankers with pairwise ranking prompting. *arXiv preprint arXiv:2306.17563*.

Zhen Qin, Le Yan, Honglei Zhuang, Yi Tay, Rama Kumar Pasumarthi, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2021. Are neural rankers still outperformed by gradient boosted decision trees? In *Proceedings of the 9th International Conference on Learning Representations*.

Walker Ravina, Ethan Sterling, Olexiy Oryeshko, Nathan Bell, Honglei Zhuang, Xuanhui Wang, Yonghui Wu, and Alexander Grushetsky. 2021. Distilling interpretable models into human-readable code. *arXiv preprint arXiv:2101.08393*.

Devendra Singh Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. *arXiv preprint arXiv:2204.07496*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023a. Is ChatGPT good at search? investigating large language models as re-ranking agent. *arXiv preprint arXiv:2304.09542*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023b. Is ChatGPT good at search? investigating large language models as re-ranking agent. *arXiv preprint arXiv:2304.09542*.

Raphael Tang, Xinyu Zhang, Xueguang Ma, Jimmy Lin, and Ferhan Ture. 2023. Found in the middle: Permutation self-consistency improves listwise ranking in large language models. *arXiv preprint arXiv:2310.07712*.

Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Neil Houlsby, and Donald Metzler. 2022a. Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131*.

Yi Tay, Vinh Q Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. 2022b. Transformer memory as a differentiable search index. In *Advances in Neural Information Processing Systems*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Paul Thomas, Seth Spielman, Nick Craswell, and Bhaskar Mitra. 2023. Large language models can accurately predict searcher preferences. *arXiv preprint arXiv:2309.10621*.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2023. A survey on large language models for recommendation. *arXiv preprint arXiv:2305.19860*.

Le Yan, Zhen Qin, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2022. Scale calibration of deep ranking models. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4300–4309.

Bianca Zadrozny and Charles Elkan. 2001. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 204–213.

Bianca Zadrozny and Charles Elkan. 2002. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 694–699.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.

Honglei Zhuang, Zhen Qin, Kai Hui, Junru Wu, Le Yan, Xuanhui Wang, and Michael Berdersky. 2023a. Beyond yes and no: Improving zero-shot llm rankers via scoring fine-grained relevance labels. *arXiv preprint arXiv:2310.14122*.

Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. 2023b. RankT5: Fine-tuning T5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2308–2313.

# A Reproducibility

## A.1 Prompts for Relevance Prediction

We used the **same prompt template for all 5 datasets** evaluated in the paper. Below is the prompt template for estimating relevance in the pseudo-rater mode:

> Passage: `{passage}`
>
> Query: `{query}`
>
> Does the passage answer the query? Output Yes or No:

## A.2 Prompts for Pairwise Preference

Below is the prompt template for pairwise preference in the pairwise ranking mode:

> Given a query `{query}`, which of the following two passages is more relevant to the query?
>
> Passage A: `{passage`$_1$`}`
>
> Passage B: `{passage`$_2$`}`
>
> Output Passage A or Passage B:

## A.3 Code and Data Release

Our experimental results are easily reproducible, using open-sourced LLMs and standard aggregation methods (win counting, sorting, and sliding window) used in the work. We intend to release pairwise preference results on all five datasets from the two open-source LLMs to aid future research. Specifically, we will release the data in JSON format, which will include query-document pair information (ids, text, label, retrieval rank and scores), along with the prompts used, the generated texts, and relevance estimation scores.

# B Computational Costs

Our constrained regression methods are based on a traditional algorithm, the extra computation cost is negligible compared with the LLM calls. Specifically, depending on the model and the token lengths of the documents, the GPU time for LLM calls to obtain one relevance estimation or one pairwise preference could vary, but it is typically on the order of 10 ms to 1 s per LLM call. For PRP, a list of 100 documents would require at least 100 s of GPU time to obtain all pairwise preferences. The constrained regression, independent of the model or the document length, can be solved (with `scipy.optimize.minimize`) in about 100 ms on common CPUs for a query of 100 documents.

# C More Results on Efficient Constrained Regression

## C.1 LLM vs non-LLM raters

A good relevance rater is important for the constrained regression methods to work. LLM pseudo-rater (PRater) scores are cheaper than the PRP scores, and are directly leveraged in our methods. On the other hand, BM25 scores are fast ad hoc results for result retrieval and are thus available at ranking stage. Here, we study the effects of replacing the LLM rater (PRater) with non-LLM rater (BM25) as the base rater for $\hat{y}$ in all constrained regression methods and as the initial ranker to select pairwise constraints in efficient sliding window (SlideWin) and top vs all pairs (TopAll) methods.

The results are summarized in Table 5. We have the following observations: First, the choice of the base rater (Base) mainly affects the relevance prediction performance: ECE of results with PRater is

Table 5: Effects of initial ranker (init) and base rater (base) on different constrained regression methods on TREC-DL 2020. Bold numbers indicate the best metrics in each column per method.

| Method | init | base | NDCG@10 | ECE |
|---|---|---|---|---|
| Allpair | - | BM25 | **0.7061** | 0.2941 |
| | - | PRater | 0.7054 | **0.0865** |
| SlideWin | BM25 | BM25 | **0.7046** | 0.2707 |
| | BM25 | PRater | **0.7046** | **0.0911** |
| | PRater | BM25 | 0.6939 | 0.2985 |
| | PRater | PRater | 0.6939 | 0.0945 |
| TopAll | BM25 | BM25 | 0.6524 | 0.5712 |
| | BM25 | PRater | 0.6938 | **0.0918** |
| | PRater | BM25 | 0.5949 | 0.3149 |
| | PRater | PRater | **0.7025** | 0.0966 |

significantly better than of those with BM25, as the relevance prediction performance of the constrained regression methods is mainly limited by the base scores $\hat{y}$. In contrast, the choice of Base is nearly insignificant to the ranking performance in AllPair and SlideWin methods, but affects ranking more in the TopAll method: TopAll with PRater Base always show better NDCG than TopAll with BM25 Base. Furthermore, the choice of the initial ranker (Init) is almost neutral on regression in terms of ECE, but has a complex effect on ranking in NDCG in SlideWin and TopAll methods. We note that using PRater as initial ranker in SlideWin leads to slightly worse NDCG than using BM25. This is attributable to the better alignment of LLM relevance rater and PRP ranker, so that the pairwise constraints become less informative than starting from initial ranking of BM25. On the other hand, using PRater as initial ranker in TopAll leads to better NDCG when PRater is the base rater and worse NDCG when BM25 becomes the Base. This is attributable to the alignment of initial ranker and base rater to select useful pairwise constraints. Based on these results, we recommend to use LLM PRater as the base rater for all constrained regression methods and use BM25 as the initial ranker for SlideWin while PRater as the initial ranker for TopAll method.

Table 6: Effects of top $k$ parameters in sliding window (SlideWin) and top vs all pair (TopAll) constrained regression methods on TREC-DL 2020. Bold numbers indicate the best metrics in each column per method.

| | | NDCG | | | | |
|---|---|---|---|---|---|---|
| Method | top $k$ | @1 | @5 | @10 | @20 | ECE |
| SlideWin | 2 | **0.8580** | 0.7367 | 0.6978 | 0.6547 | 0.0966 |
| | 5 | **0.8580** | **0.7535** | 0.7013 | **0.6698** | 0.0936 |
| | 10 | **0.8580** | **0.7535** | **0.7046** | 0.6674 | 0.0911 |
| | 20 | **0.8580** | **0.7535** | **0.7046** | 0.6676 | **0.0890** |
| TopAll | 2 | 0.7778 | 0.7014 | 0.6762 | 0.6366 | 0.0981 |
| | 5 | **0.8642** | 0.7319 | 0.6965 | 0.6559 | **0.0954** |
| | 10 | 0.8549 | **0.7367** | **0.7025** | **0.6593** | 0.0966 |
| | 20 | 0.7685 | 0.7052 | 0.6848 | 0.6520 | 0.0987 |

### C.2 Choice of parameter k

We investigate the effect of hyper-parameter $k$ in both SlideWin and TopAll methods. Note that though we have chosen the same character $k$ to represent the parameters, the actual meanings of the parameters are different in the corresponding methods: top $k$ is the number of top results to be sorted in the SlideWin, and $k$ is the number of the top results in the initial ranker to fetch pairwise constraints.

In Table 6, primarily, we find the choice of top $k$ affects the ranking performance (NDCGs) only. In specific, ignoring numerical fluctuations, increasing parameter $k$ of SlideWin monotonically improves

NDCG@$m$ till $k \sim m$. On the other hand, increasing parameter $k$ of TopAll leads to non-monotonic NDCG@$m$ that is optimized approximately around $k \sim m$. The intuition of the difference between SlideWin and TopAll is that (1) the parameter $k$ of SlideWin is the top number after pairwise ordering, so that top $k$ result orders will always be consistent with PRP results so as NDCG@$m$, as long as $k > m$; (2) while the parameter $k$ of TopAll is the number of top results in initial ranker, which is different from the PRP results, so that when $k < m$, increasing $k$ is likely improving NDCG@$m$ as more top results are included, however, when $k > m$, more intra-top pair constraints become more dominant than top vs rest pairs, which may break the order between top $k$ vs rest results and lead to worse NDCG.

Table 7: Consolidation results of listwise ranking on TREC-DL 2019 and TREC-DL 2020. ListRank method reranks the top 20 results retrieved from BM25 and the top 20 results from PRater. Allpair method is then applied to consolidate ListRank and PRater predictions. Bold numbers indicate the best metrics in each row.

| Dataset | Metric | PRater | BM25 Top20 | | PRater Top20 | |
|---|---|---|---|---|---|---|
| | | | ListRank | Allpair | ListRank | Allpair |
| TREC-DL19 | NDCG@10 | 0.6461 | 0.6379 | 0.6567 | **0.7477** | **0.7477** |
| | ECE | 0.1167 | 0.1614 | **0.1149** | 0.1549 | 0.1237 |
| | MSE | 0.0688 | 0.2008 | **0.0660** | 0.1586 | 0.0711 |
| TREC-DL20 | NDCG@10 | 0.6539 | 0.6123 | 0.6442 | **0.6694** | **0.6694** |
| | ECE | 0.0991 | 0.1309 | 0.0988 | 0.1291 | **0.0963** |
| | MSE | 0.0632 | 0.1786 | 0.0618 | 0.1462 | **0.0596** |

## D Applying Consolidation Methods to Listwise Ranking

Our consolidation methods are applicable to the LLM-based listwise ranking. In Table 7, we summarize our results of the consolidation method (Allpair in specific) applied to the ListRank, our reproduction of the RankZephyr approach (Pradeep et al., 2023) on the PaLM 2 model (Google et al., 2023).

In ListRank, we train an LLM to directly predict the final ranking order of top 20 retrieved candidates. In specific, we have compared top 20 candidates retrieved with BM25 score (BM25 Top20) and with an LLM PseudoRater (UL2, PRater Top20 in Table 7). As a validation of our reproduction, the NDCG@10 of ListRank on BM25 Top20 is comparable to the value in Table 5 in the RankZephyr paper (Pradeep et al., 2023).

The NDCG metrics are measured with the predicted order of the top 20 results. The ECE and MSE metrics are computed on scaled ranking scores from the predicted ranks $r_i$:

$$s_i = \frac{1}{20} \max(0, 21 - r_i).$$

The "Allpair" columns next to the "ListRank" columns show our consolidation results with all pairwise order constraints of top 20 results from the ListRank predictions. In all consolidation results, the scores are computed with the PRater scores as initial scores.

As shown in Table 7, Allpair methods outperform both PRater and ListRank baselines in both ranking and relevance prediction. These results verify the generalizability and efficacy of our proposed method.